key accelerator as it normally would by simply referencing the appropriate RSA private key stored in the stateless module.

[0202] The application key cache **1104** also may store key material that may be used by external cryptographic acceleration processors. For example, the cache **1104** may store decrypted application keys (e.g., the RSA private key for an application executing on the device that contains the stateless module).

[0203] The stateless module enforces key policy for keys used within the remote client. The key policy may be set by the key manager for all keys that are delivered to the stateless module. The key policy indicates how the key can be used by the stateless module. In addition to usage policy, the stateless module can enforce a lifetime for keys. Typically, a key lifetime is a relative time from the time at which the key is loaded into the stateless module. The key manager can use the multiple levels of key hierarchy and the lifetime policy enforcement to ensure that keys are used properly and are revocable at the stateless module.

[0204] A security assurance logic block **1128** protects the stateless module from system security attacks. To this end, several system monitors may be coupled with the other components in the stateless module and/or the chip (and/or the system) within which the stateless module resides.

[0205] In some embodiments, the protection circuits trigger a reset of the stateless module when an attack is detected. This reset may wipe out all transient information in the stateless module. For example, all key cache locations may be cleared. An interrupt may be provided to the local host with information on which protection mechanism triggered the reset.

[0206] A low frequency protection circuit ensures that the operating frequency of the stateless module does not fall below given threshold. This ensures that the time tick register value can not be compromised within the limit of a reference frequency. In addition to protecting the time tick value, the low frequency protection circuit makes it more difficult to implement successful dynamic attacks that attempt to read values within the stateless module while it is operating. In this case, the higher the threshold value, the better protection that is provided.

[0207] An operating point protection circuit may be provided to ensure that all logic within the stateless module operates as designed for all process, voltage and temperature conditions (or across all operating points). The protection circuit helps ensure that an attacker cannot change the operating point such that a timing path is violated in the stateless module.

[0208] A watchdog timer block may be used during processing to ensure that command execution completes within an expected period of time. The timer is set by the master controller whenever a command (or sub-command such as a public key operation) is started. The set time is based on the expected maximum command length. If the watchdog timer reaches zero a reset is issued to the stateless module. The watchdog timer cannot be turned off and must be written periodically by the master controller to avoid clearing the stateless module. The watchdog timer may be frozen when the stateless module is taking command input from the host.

[0209] A reset monitor provides protection against multiple reset attacks. The reset monitor uses a timer based on the time tick register increment that requires at least one tick before allowing more than, for example, sixteen resets to occur. If more than sixteen resets occur within the time tick, the stateless module will require at least two time ticks before releasing the sixteenth reset. The reset protection is disabled until the NVM has been properly programmed. For example, it may be disabled during manufacturing tests.

[0210] A hardware protection mechanism may be provided for entering and exiting a secure state while the stateless module transitions between enabling/disabling the external interface. The stateless module boots to a secure state with the external interface disabled. That is, the interface is locked out by hardware. Once reset processing and self-tests have completed, the master controller sequences through a series of commands to exit the secure state and enter a USER state. In some embodiments these commands require execution of a predefined set of sequential instructions be written to non-sequential addresses.

[0211] The hardware tracks the number of clocks it takes to execute each step of the sequence and ensures that these commands occur in the required order to the required address at exactly the right clock cycle. After the exit logic has completed, the mode is set via hardware to USER mode. In USER mode, the hardware locks out master controller access to all of the internal blocks except the data buffer and the data input/output registers (only blocks that are required to move data into the device).

[0212] Once the command has been moved into the data buffer, the master controller sequences a series of commands to return to the secure state. This sequence is again tracked and enforced via the hardware block to enter into secure mode. It also ensures via hardware that the master controller enters the secure mode with the proper entry address.

[0213] The master controller ROM **1108** may be programmed using an extra bit to indicate which instructions are valid code entry and code exit points. The instruction code entry/exit points are enforced in hardware whenever the master controller takes a non-sequential code fetch. This mechanism helps to ensure that it will be difficult for an attacker to get the master controller to bypass certain portions of code. As a result, it may be virtually impossible to successfully attack the module by causing random jumps in the program execution.

[0214] To reduce cost and die space, the stateless module may not handle processing related to communication protocols. Instead, the requirements of communication protocols may be handled by an associated device driver (or integrated processor).

[0215] In an alternative embodiment, the stateless module may be assigned long-term keys. In this case, the stateless module may not need to interface with a head-end server (e.g., key manager).

[0216] Referring now to **FIG. 12**, an example of operations that may be performed by one embodiment of a stateless module will be discussed. As represented by block **1202**, when the stateless module is initialized for the first time after manufacture (e.g., during final test of the chip), the master controller may cause the random number generator